Programming Dedicated Microprocessors

Macmillan Computer Science Series

Consulting Editor

Professor F.H. Sumner, University of Manchester

S.T. Allworth, *Introduction to Real-time Software Design*
Ian O. Angell, *A Practical Introduction to Computer Graphics*
R.E. Berry and B.A.E. Meekings, *A Book on C*
G.M. Birtwistle, *Discrete Event Modelling on Simula*
T.B. Boffey, *Graph Theory in Operations Research*
Richard Bornat, *Understanding and Writing Compilers*
J.K. Buckle, *Software Configuration Management*
W.D. Burnham and A.R. Hall, *Prolog Programming and Applications*
J.C. Cluley, *Interfacing to Microprocessors*
Robert Cole, *Computer Communications, second edition*
Derek Coleman, *A Structured Programming Approach to Data*
Andrew J.T. Colin, *Fundamentals of Computer Science*
Andrew J.T. Colin, *Programming and Problem-solving in Algol 68*
S.M. Deen, *Fundamentals of Data Base Systems*
S.M. Deen, *Principles and Practice of Database Systems*
P.M. Dew and K.R. James, *Introduction to Numerical Computation in Pascal*
M.R.M. Dunsnuir and G.J. Davies, *Programming the UNIX System*
K.C.E. Gee, *Introduction to Local Area Computer Networks*
J.B. Gosling, *Design of Arithmetic Units for Digital Computers*
Roger Hutty, *Fortran for Students*
Roger Hutty, *Z80 Assembly Language Programming for Students*
Roland N. Ibbett, *The Architecture of High Performance Computers*
Patrick Jaulent, *The 68000 – Hardware and Software*
J.M. King and J.P. Pardoe, *Program Design Using JSP – A Practical Introduction*
H. Kopetz, *Software Reliability*
E.V. Krishnamurthy, *Introductory Theory of Computer Science*
V.P. Lane, *Security of Computer Based Information Systems*
Graham Lee, *From Hardware to Software – an introduction to computers*
A.M. Lister, *Fundamentals of Operating Systems, third edition*
G.P. McKeown and V.J. Rayward-Smith, *Mathematics for Computing*
Brian Meek, *Fortran, PL/I and the Algols*
Barry Morrell and Peter Whittle, *CP/M 80 Programmer's Guide*
Derrick Morris, *System Programming Based on the PDP11*
Pim Oets, *MS-DOS and PC-DOS – A Practical Guide*
Christian Queinnec, *LISP*
W.P. Salman, O. Tisserand and B. Toulout, *FORTH*
L.E. Scales, *Introduction to Non-linear Optimization*
Peter S. Sell, *Expert Systems – A Practical Introduction*
Colin J. Theaker and Graham R. Brookes, *A Practical Course on Operating Systems*
J-M. Trio, *8086–8088 Architecture and Programming*
M.J. Usher, *Information Theory for Information Technologists*
B.S. Walker, *Understanding Microprocessors*
Peter J.L. Wallis, *Portable Programming*
Colin Walls, *Programming Dedicated Microprocessors*
I.R. Wilson and A.M. Addyman, *A Practical Introduction to Pascal – with BS6192, second edition*

Programming Dedicated Microprocessors

Colin Walls

M
MACMILLAN

© Colin Walls 1986

All rights reserved. No reproduction, copy or transmission of this publication may be made without written permission.

No paragraph of this publication may be reproduced, copied or transmitted save with written permission or in accordance with the provisions of the Copyright Act 1956 (as amended).

Any person who does any unauthorised act in relation to this publication may be liable to criminal prosecution and civil claims for damages.

First published 1986

Published by
MACMILLAN EDUCATION LTD
Houndmills, Basingstoke, Hampshire RG21 2XS
and London
Companies and representatives
throughout the world

British Library Cataloguing in Publication Data
Walls, Colin

Programming dedicated microprocessors.—
(Macmillan computer science series)

1. Microprocessors—Programming

I. Title

005.2'6 QA76.6

ISBN 978-0-333-40952-7

ISBN 978-1-349-08594-1 (eBook)

DOI 10.1007/978-1-349-08594-1

To my father
Lawrence E. Walls, FRIBA, FIAA

Contents

<i>Preface</i>	<i>x</i>
<i>Acknowledgements</i>	<i>xii</i>
1 Introduction	1
1.1 Firmware Development	1
1.2 Hardware Tradeoff	2
1.3 Using this Book	3
2 Development Techniques	4
2.1 Introduction	4
2.2 Development Systems	6
2.3 Code Assembly	7
2.4 Integration and Testing	7
2.5 Production	11
3 Aspects of Hardware Design	13
3.1 Introduction	13
3.2 Logic Circuits	13
3.3 Addressing	19
3.4 Memory	24
3.5 Input/Output	26
4 Interrupts	29
4.1 Introduction	29
4.2 Types of Interrupt	29
4.3 Interrupt Control	30
4.4 Interrupt Service Routines	32
4.5 Applications of Interrupts	33
5 Z80 Interrupts	39
5.1 Introduction	39
5.2 Maskable Interrupts	40
5.3 Non-maskable Interrupts	41
5.4 Hardware Facilities	42
5.5 The NSC800	44

6	Input/Output	45
6.1	Introduction	45
6.2	I/O Addressing	46
6.3	Parallel I/O	47
6.4	Serial I/O	55
6.5	Bulk Data Transfer	65
6.6	Generalised I/O	66
6.7	Implied Output	69
7	Multi-tasking	70
7.1	Introduction	70
7.2	Approaches to Multi-tasking	70
7.3	Executive Implementation	71
7.4	Programming for Multi-tasking	75
8	Implementing a Real-time Multi-tasking Executive	78
8.1	Introduction	78
8.2	Task Swapping	79
8.3	Procedures	80
8.4	Building and Using the Executive	87
8.5	Efficiency	92
8.6	Other Features	94
9	High-level Languages	97
9.1	Introduction	97
9.2	BASIC	97
9.3	Compiled Languages	99
9.4	Threaded Interpretive Languages	100
9.5	Cross-compilation	102
10	Diagnostics	103
10.1	Introduction	103
10.2	Self-testing	103
10.3	Status Displays	108
10.4	Watchdogs	110
10.5	Interactive Diagnostics	111
11	Communications: a Firmware Application	113
11.1	Introduction	113
11.2	Communications Layers	113
11.3	Physical Link Layer	115
11.4	Data Link Layer	121
11.5	Network Layer	127

CONTENTS

ix

<i>Appendix A: Structured Design Code</i>	<i>128</i>
A.1 Pseudo-code	128
A.2 SDC Syntax	128
 <i>Appendix B: Hexadecimal Codes</i>	 <i>132</i>
B.1 Intel Format	132
B.2 Motorola Format	134
B.3 Tektronix Format	135
 <i>Appendix C: Z80 Microprocessor</i>	 <i>137</i>
C.1 Introduction	137
C.2 Z80 Architecture	138
C.3 Z80 Instruction Set	139
 <i>References</i>	 <i>185</i>
 <i>Index</i>	 <i>188</i>

Preface

Some years ago I was given the task of writing some software for a communications controller, based upon an 8085 microprocessor. This required me to learn the assembly language for this processor, which, given the limited instruction set, was not difficult. I also needed to understand the application, and this was achieved by an iterative agreement of the specification with the customer. There remained, however, one problem: how did the microprocessor interact with the rest of the hardware to perform the required task?

Long before I learned anything about computers, I was interested in electronics. This past interest provided the starting point for bringing the software and hardware together and solving my problem. It was simply a matter of asking the right questions, looking at some circuit diagrams and making a few informed guesses. This approach must have been successful, as the project was completed without any fundamental rewrites being necessary.

As time went by, I encountered other software engineers with the same sorts of problem. Frequently, however, their background was *purely* software, so that even basic electronic principles were unknown territory. I hope I helped by answering some of their questions, but I could not suggest any appropriate books from which to glean sufficient knowledge of electronics to program dedicated microprocessors, without extraneous detail. This book is an attempt to fill that gap.

The principal aim of this book is to guide the programmer's transition from 'software' to 'firmware'. In this context, 'firmware' is defined as the controlling program(s) of a dedicated microprocessor. Most programmers acquire their initial experience using 'real' computers (that is, minis or mainframes) or personal microcomputers. In either case, programming is carried out under the control of system software. Whether it is a simple BASIC interpreter or a full-blown multiprocessing operating system is unimportant; the change to the 'unsupported' environment of dedicated microprocessors may be quite traumatic. On the whole, the programmer has more things to think about: interrupts, input/output, timing, scheduling. It is these techniques, together with those used for firmware development, which are described.

The assumption is made that dedicated microprocessors are generally eight-bit devices. This is currently the case, but the techniques are still applicable to larger processors, as they become economically viable. The Z80 is used as an example, since it is the most popular eight-bit device and should be readily understood. A

general knowledge of assembly language programming is assumed, along with some familiarity with the Z80 instruction set. Although primarily directed at practising software engineers, the material should be of equal interest to students of programming.

Acknowledgements

There are many people, too numerous to mention individually, who have helped me with the progress of my career (and, hence, this book), to whom I am grateful.

For specific help with this book, I would particularly like to thank Robert Day, who gave up many potential drinking hours to review an early typescript, and Ian Morte, who asked many questions, some of which were quite sensible, that I hope are answered here.

Lastly I would like to thank my wife, Linda, for helping prepare the typescript and putting up with me without complaining (much).

Wokingham, 1986

COLIN WALLS